



LINUX Expert

Internal telephone exchanges are no longer just available to large corporations. **Daniel Goscomb** shows you how to create a private branch exchange for a business of any size, absolutely free



CHECK POINT TO ACQUIRE SOURCEFIRE

Security company Check Point Software Technologies has announced plans to acquire Sourcefire, the company behind the popular Snort open-source intrusion detection system.

This news has caused some users to worry that Snort will cease to be available under the GPL after the acquisition. Sourcefire's chief executive officer, Wayne Jackson, addressed these fears, saying that the Snort community provides "a huge network of security expertise" from which the company can learn and that it had no reason not to carry on with the open-source development of the product.

The current source code is available under the GPL so if Check Point were to start work on a closed version, the project could easily be forked and worked on by independent developers.

■ OSDL TO FORM MOBILE TASKFORCE

Open Source Development Labs is creating a taskforce to encourage developers to optimise Linux for handheld and mobile devices. As well as optimising the Linux kernel to run on mobile phones, the initiative will attempt to spur the development of applications that run on the platform.

The initiative includes the likes of MontaVista Software, Motorola, PalmSource, Trolltech and Wind River. Motorola is the largest supplier of Linux mobile devices, having shipped more than three million units to date.

■ MANNHEIM SWITCHES TO LINUX

The German city of Mannheim is hoping to complete phase one of its migration to Linux by the end of 2005. In the first phase, Windows NT servers will be replaced with Linux machines. The next phase will include switching around 3,700 desktops over to Linux.

Using IBM as a technical partner, the city hopes to be free of Microsoft and is currently evaluating OpenOffice.Org as an alternative to the Microsoft Office suite. Munich is also due to migrate to Linux in mid-2006.

Businesses spend a fortune on telephone services. Calls abroad and telephone line rentals all add up, so large companies save money by using a PBX (private branch exchange). By running their own internal exchanges, they don't need to rent a line for every employee, and people inside the company can call each other without incurring the costs of making calls over the outside telephone system. You can have 500 telephones on desks in the building but only, say, 20 real telephone lines coming in.

A PBX can be connected to a Voice over IP (VoIP) service to provide cheaper calls too, and it can also manage voicemail and Interactive Voice Response menus that allow you to guide callers with messages such as "Press one for sales; press two for technical support" and so on. You can also set up phone conferences, place callers on hold with music and transfer calls from one desk to another. Even very small companies can benefit from using a PBX, although buying one can be prohibitively expensive. We'll show you how to slash the expense by turning a PC into a PBX using free software and some inexpensive, optional hardware.

The software we're using is called Asterisk, and you can download it for free from www.asterisk.org. Asterisk is sponsored by a company named Digium, which was formed by Asterisk's principal author. Digium also manufactures compatible telephony interfaces, such as analogue and ISDN cards. Although you can set up a basic PBX without any additional hardware, if you want to plug in regular analogue phones and use the BT network, you'll need a card such as Digium's Wildcard TDM400P, which is the one we've used in this article. It's available from Digium's US website at www.digium.com or, if you prefer to buy from the UK, www.voiptalk.org. The Wildcard TDM400P costs around £230 including VAT.

In this article we'll show you everything you need to know to set up a telephone network that connects an external BT line to two analogue telephones, one or more SIP phones and one or

more software phones running on desktop PCs. The system will also use a VoIP service to reduce outgoing call costs, and we'll show you how to install a great voicemail system too. We'll even show you where to start if you want to upgrade to more external lines.

THE HARDWARE

The minimum hardware you need to set up an Asterisk PBX is a PC. All other items are optional extras, as you can use a software phone on your PC to make and receive calls. We're opting to use Digium's Wildcard TDM400P (model TDM22B), which has ports for two internal extensions and two incoming lines. To make full use of the card you'll also need two analogue phones. If you want to use hardware or software VoIP SIP phones, you'll need a network card in the PC too.

To hook up the hardware, first plug the PCI card into a spare slot in the PC. This card also has a power socket similar to those found on IDE hard disks. Plug in a spare 12-volt power cable so that the card can provide power to the analogue phones. Put the cover back on your computer and examine the ports on the back of the new card.

The TDM400P has two types of port. The foreign exchange station (FXS) ports are for internal lines, while the foreign exchange office (FXO) ones are for external lines. They are numbered one to four, and ports one and two are the FXS (internal) ports. Ports three and four are the FXO ports for external lines, and we need only one of these – port three. Use a phone cable to connect port three to the phone socket in the wall. You'll need to use a line adaptor to convert the PBX end into an RJ45 connection.

Plug analogue phones directly into ports one and two on the TDM400P. You will probably also need some line adaptors (available from RS, www.rswww.com, stock number 208-1786) to convert the phone's plug into an RJ45 connector. Having RJ45 connections might seem inconvenient, but if you ever set up your PBX system in a building

TELEPHONE JARGON EXPLAINED

Integrated services digital network (ISDN)	A type of communications line capable of supporting two or more voice calls.
Foreign exchange office (FXO)	An interface on a PBX that connects to the outside telephone system.
Foreign exchange station (FXS)	An interface on a PBX that connects directly to your analogue phones and faxes.
Private branch exchange (PBX)	A PBX system shares one or more outside lines with a group of users and allows its users to call each other without connecting to an outside line.
Primary rate interface (PRI)	A 30-channel ISDN system for large organisations. Basic Rate Interface systems allow only two channels for calls.
Public switched telephone network (PSTN)	A main telephone network, such as the one run by BT.
Session Initiation Protocol (SIP)	A protocol that allows telephony devices to work over IP networks such as the internet and computer networks in offices.
Trunk	A large communications channel that can handle multiple connections or calls.
Voice over IP (VoIP)	VoIP services provide telephone connections across IP networks. Used as an alternative to PSTN to save money.

that has professional wiring in place, you'll be very grateful for them.

INSTALLING THE SOFTWARE

To get the PBX running you will need the following software packages: ncurses; openssl; zlib; bison; mpg123. The last one is optional, but you'll need it if you want to have music playing when callers are on hold. These should be installed by default on most Linux distributions. You'll also need zaptel and Asterisk, which we'll download and install.

The best way to obtain Asterisk is from Digium's Concurrent Versions System (CVS) repository. This allows us to download and build the latest beta branch of Asterisk. We have chosen this release as it has extra features and a host of bug fixes that are not available in the 1.0 version.

Change to a directory that will hold the source code. We will use /usr/src, so type:

```
# cd /usr/src
```

We must now set the options for CVS to tell it which server we wish to log into, as well as our login credentials. The password is anoncvs. Run the following commands:

```
# export CVSROOT=:pserver:
# anoncvs@cvs.digium.com:/usr/cvsroot
# cvs login
```

We now need to tell CVS to get the two required packages, zaptel and Asterisk, as well as a library called libpri:

```
# cvs checkout -r v1-2-0-beta1 zaptel
# asterisk libpri
```

You will now see that CVS downloads each required file for you. The zaptel package contains the drivers for the Digium hardware, while the libpri libraries are used by Asterisk. Although in our small PBX scenario we are not using a Primary Rate Interface (PRI) connection (see the Telephone Jargon box opposite), the library is still required by Asterisk. Issue the following commands:

```
# cd /usr/src/zaptel
# make clean; make install
# cd /usr/src/libpri
# make clean; make install
```

Once you have successfully installed these, you need to install Asterisk:

```
# cd /usr/src/asterisk
# make clean; make install
```

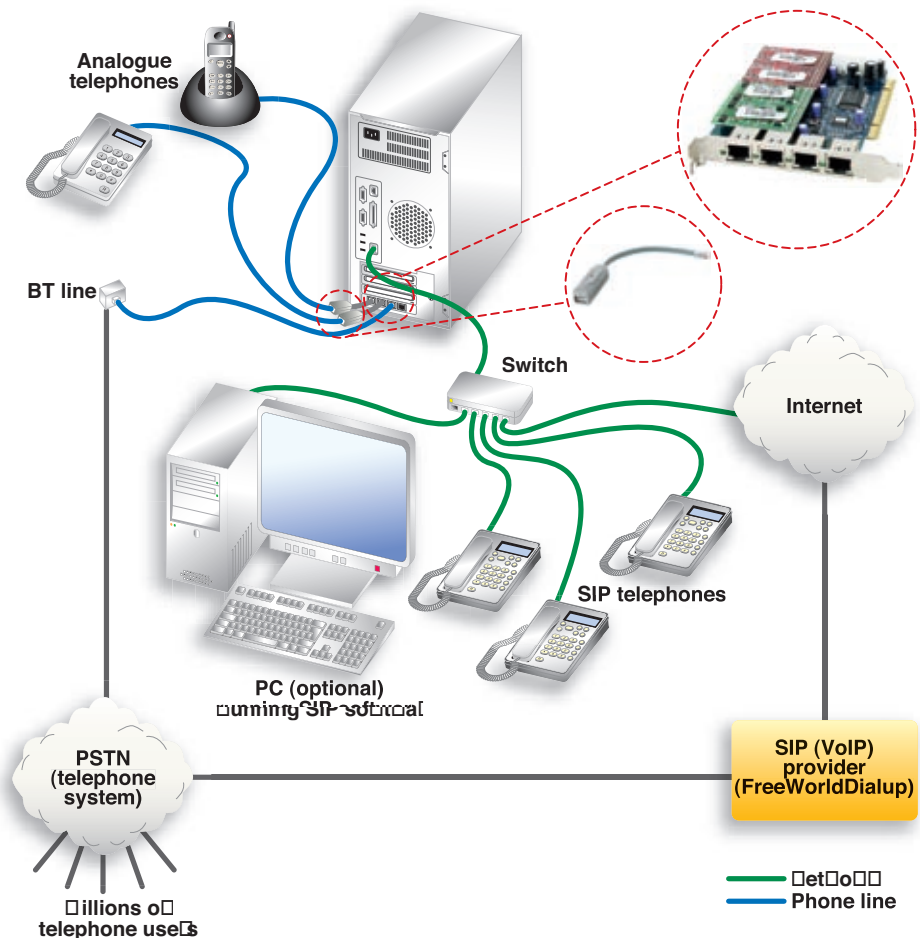
Asterisk is now installed, but there are no configuration files, which you need to install separately. Asterisk has lots of features, so if you are unfamiliar with the program, it is advisable to install the sample configuration files and edit them rather than creating them from scratch. To install the sample files and also the documentation, issue the following commands:

```
# make samples
# make progdocs
```

To be able to install the documentation you also need a program called doxygen (www.doxygen.org)

CONNECTING THE PHONE SYSTEM

Run a combination of analogue and SIP phones alongside a regular telephone line and an optional VoIP service. You can even use a PC as a phone if you want.



installed on your system. Asterisk, its configuration files and its documentation should now be installed on your system. All Asterisk's configuration files are stored in the directory /etc/asterisk. The zaptel hardware drivers have just one configuration file, which is located at /etc/zaptel.conf. You need to configure this before it will work, though.

The first thing to do is tell the zaptel drivers about the channels available on our hardware. To do this we need to edit the file /etc/zaptel.conf so it looks like this:

```
fxoks=1-2
fxsks=3-4
loadzone = uk
defaultzone=uk
```

This tells the zaptel driver to act as an FXO on ports one and two and send out calls to the phones. It will also act as an FXS on ports three and four to receive calls from the Public Switched Telephone Network (PSTN). This might seem confusing because earlier we said that ports one and two were the FXS ports, which are used for internal lines. The server software operates as an FXO, but the card works as an FXS. You don't need to understand how FXOs and FXSs work to build a PBX system, but if you are interested in finding out

more, visit the website at www.digium.com/index.php?menu=fxsvfxo.

You now need to load the kernel module that came with the software and pass the configuration to it:

```
# modprobe wctdm
# ztcfg
```

If running the ztcfg command generates any text on the screen, you have a problem in your zaptel.conf file and should go back and check it. Now that the system knows about the Digium card, we must also let Asterisk know it is there. To do this, open the /etc/asterisk/ file and edit it so that it looks like the following:

```
[channels]
language=en
rxgain=0.0
txgain=0.0
immediate=no
echocancel=yes
echocancelwhenbridged=yes
echotraining=yes
signalling=fxo_ks
context = dialphone
channel => 1,2
```



```
signalling=fxs_ks
context=incomingfrompstn
channel => 3,4
```

This file tells Asterisk how to interpret calls on these ports. The parameters are grouped according to channel. Any commands written above a channel => line relate to that channel or group of channels. These settings are also inherited by the next channel group, unless this later group has similar parameters but different settings. For example, here we have enabled echo training on all channels, setting the gains to 0 and the language to English. Although these settings are bundled in the group related to channels one and two, channels three and four will inherit these too. They won't inherit the context setting of fxs_ks because they have their own setting of incomingfrompstn.

The context settings relate to groups of dialling commands that are placed in the extensions.conf file, which is also called a dialplan. We have set ports one and two to have FXO signalling and be placed in the dialling context dialphone, and set ports three and four to have FXS signalling and be placed in the incomingfrompstn context.

Our third internal phone is a Session Initiation Protocol (SIP) client, which is essentially a phone attached directly to the network using its own network connection. We must add an account for it so that it can authenticate to our Asterisk server and make and receive calls. To do this, open the sip.conf file and add an entry to the bottom of the file that reads:

```
[sipphone]#
type=friend#
username=sipphone#
secret=mysecret#
host=dynamic#
canreinvite=no#
disallow=all#
allow=gsm#
allow=ulaw#
context=dialphone#
```

You can use a hardware SIP phone such as the Budgetone, which costs less than £50, or, if you prefer, configure a software phone on a PC.

Now we have our three phones connected, we need to create a basic dialplan so they can actually make and receive calls. To do this, create a new extensions.conf file. Make a copy of the original, as the examples it contains could prove useful later. Start with a very basic configuration, enough to place calls between the internal analogue and SIP phones. Edit the file and enter the following:

```
[general]#
static=yes#
writeprotect=yes#
autofallthrough=yes#
```

```
[globals]#
CONSOLE=Console/dsp#
```

```
[phones]#
exten => 6000,1,Dial(Zap/1,20,tTr)#
exten => 6001,1,Dial(Zap/2,20,tTr)#
exten => 6002,1,Dial(SIP/
sipphone,20,tTr)#
```

```
[dialpstn]#
ignorepat => 9#
exten => _9.,1,Dial(Zap/3/${EXTEN:1})#
```

```
[dialphone]#
include => phones#
include => dialpstn#
```

Each configuration block except general and globals contains commands that the Asterisk server will follow as and when it matches a command. Each section starts with a [title] and is called a context. Within a context you may define extensions using the exten identifier, and include other contexts with the include identifier.

The dialplan will process a phone call as specified by the channel's configuration. All our phones have been configured with the line context=dialphone, which means they will start in the dialphone context in the extensions.conf file. This context includes the phones and dialpstn contexts. As such, the system steps through each rule until it finds a match.

When a match is found the system follows a set of commands. For example, in the dialphone context, as soon as a 9 is dialled, it matches the exten => _9. command in the dialpstn context, which passes the dialled number (minus the initial 9) to the phone line that's plugged into port three on the Digium card. This means that dialling 9 on an internal phone lets you make calls to external lines. If you dial 6000 without adding a preceding 9, the phone plugged into port one on the Digium card will ring.

The command to ring a channel is Dial. This command uses parameters such as channel, ring time and options. The channel is a port to which the phone we want to ring is attached. If it's a line connected to another network (such as the PSTN), we also add the outgoing number to this value. The ring time is the length of time in seconds that it must ring the channel before giving up. In the example above, this is set to 20 seconds. There are plenty of other options, too. Here we have used t to allow the called user to transfer calls, T to allow the caller to transfer the call and r to generate a ringing tone for the calling user.

INCOMING CALLS

At this stage our PBX allows calls between the extensions and calls to the public telephone network, but it can't yet handle incoming calls. As we have only a single telephone line with one number, we need to set the system up to ring all the phones at once until one of them is picked up. Alternatively you could set up Asterisk to answer the call for you and present the caller with a menu asking them which desk they require, or even present them with a company directory where they enter the first three letters of the first or last name of the person to whom they wish to speak to be put through to them. This is a very useful feature for large installations.

We have specified that incoming calls will be handled by the incomingfrompstn context. This is just another context in the extensions.conf file, so we must add it there. In this example it is simple, and consists of the following:

```
[incomingfrompstn]#
exten => s,1,Dial(Zap/1&Zap/2&SIP/
```

```
sipphone,20,tTr)#
```

Here we use the Dial command to ring more than one phone at a time. The phone entries are separated by ampersands and the options regarding ring time, transferring and ring tones are added at the end. Now that the basic configuration is in place, you can fire up Asterisk and try it out. To start Asterisk use the command:

```
# asterisk -cgvvv#
```

This starts Asterisk in a console mode with the extra verbosity setting, which shows extra debug information. If Asterisk fails to start, there is an error in the configuration files. If you look at the information from the bottom up, you should be able to see the error.

Once Asterisk has started, make some calls. Pick up the phone that's plugged into port one on the Digium card and dial 6001. This should ring the other phone. Get a friend to pick it up and check that you can speak to each other. Next, try to make an outside call. Dial 9 followed by the number you want to reach. For example, to call BT sales you would dial 90800 800152. You should hear ringing and then the other end pick up, just as if you had dialled out on your normal phone line.

The last test is to dial your phone number, the line into which you have plugged the Asterisk box, from the outside to see if all the phones ring. If everything works, congratulations. You have a basic, working PBX.

ADDING TO YOUR PHONE SYSTEM

Now that we have a working phone system it's time to add some features to it. Voicemail is a very useful facility, and adding it takes only two steps. First, we need a setting that puts the caller through to voicemail when their call is unanswered. Second, the system should enable users to call in and listen to their voicemail.

To set up three voicemail boxes, one for each extension, edit the voicemail.conf file and change it so that it reads:

```
[general]#
serveremail=asterisk@yourdomain.com#
attach=yes#
maxsilence=10#
```

```
[default]#
6000 => 6000,User
1,user1@yourdomain.com#
6001 => 6001,User
2,user2@yourdomain.com#
6002 => 6002,User
3,user3@yourdomain.com#
```

The last three lines set up the three voicemail boxes. Each line is formatted like this:

```
Mailbox number => PIN for logging in,
user's name, user's email address
```

You need to make an addition and two small changes to the extensions.conf file. First, add a new context called services with the settings as below:

```
[services]#
exten => 8000,1,Ringing#
```




```
exten => 8000,2,Wait(2)
exten => 8000,3,VoiceMailMain
```

This adds the services context with one extension number (8000). You can see from this that each extension may have multiple steps. In this example there are three. Each command is executed in time. Next, add the following line to the dialphone context to allow the phones to access voicemail when dialling 8000:

```
include => services
```

Finally, instruct the dialplan to go to voicemail when the phone is busy or unanswered. Change the phones context so it reads:

```
[phones]
exten => 6000,1,Dial(Zap/1,20,tTr)
exten => 6000,2,VoiceMail,u6000
exten => 6000,102,VoiceMail,b6000
exten => 6001,1,Dial(Zap/2,20,tTr)
exten => 6001,2,VoiceMail,u6001
exten => 6001,102,VoiceMail,b6001
exten => 6002,1,Dial(SIP/sipphone)
exten => 6002,2,VoiceMail,u6002
exten => 6002,102,VoiceMail,b6002
```

These extra commands will ask the caller to leave a voicemail message, first playing either a busy or unavailable message depending on the command. The u command means unavailable, b is for busy. The numbers following the extension, such as 1,2 and 102, are the priorities of the rules. Priority 1 rules are checked first.

Here is how the PBX processes a call using the rules above. Imagine someone has called the 6000 extension. The 6000 extension is linked to the Dial Zap/1 phone. If 20 seconds pass and no-one picks up the phone, the system moves on from the first Dial line, increasing the value of the priority by one. This makes a priority of two. The next line has a priority of two, so the PBX uses this command, which enters the voicemail application and plays the unavailable message.

If the line is engaged the system moves on from the Dial rule and adds 101 to the priority value, making a total of 102. This corresponds with the third line, which enters the voicemail application and plays the busy message.

Restart Asterisk and test the voicemail system. Dial 8000 from an internal phone to set up your voicemail box and check your messages. Then try calling another extension, let it ring for 20 seconds and see what happens.

SAVING MONEY WITH VoIP

Our PBX system already has a VoIP (SIP) extension, so why not add a SIP external connection to save money on calls? There are many VoIP providers, all of which offer different services. To demonstrate this in action we've set up a FreeWorldDialup account, which allows us to call any other user on the FWD network for free. It also lets us make calls to freephone numbers in many countries around the world. Because it's free, FWD is a good service to practise with. You could use Vonage or another VoIP service if you wanted.

You need to sign up for your own FWD account by registering at www.pulver.com/fwd. When you register you are given an FWD number. This is your phone number and username. First, you

need to set up the channel on the PBX so that it can make and receive calls with the service. Edit the sip.conf file and add two sections. First, add the line:

```
register => FWDNUM:
PASSWD@fwd.pulver.com/FWDNUM
```

to the [general] section near the top of the file, replacing FWDNUM with your number and PASSWD with your password. Then, at the end of the file, add the following:

```
[fwd]
type=friend
secret=PASSWD
username=FWDNUM
fromuser=FWDNUM
fromdomain=fwd.pulver.com
host=fwd.pulver.com
dtmfmode=inband
nat=yes
canreinvite=no
context=incomingfrompstn
```

Again, replace FWDNUM with your number and PASSWD with your password. To be able to make calls out on this number we need to add the following to the dialpstn context in extensions.conf:

```
exten => _5.,1,Dial(SIP/${EXTEN:
1}@fwd,70)
```

This means that any number preceded by a 5 will be sent out over the FWD network. Reload Asterisk and enter the number 5613 to try out the echo test. If that works, you should be able to call other FWD subscribers. Follow the same procedure to install a subscription-based service.

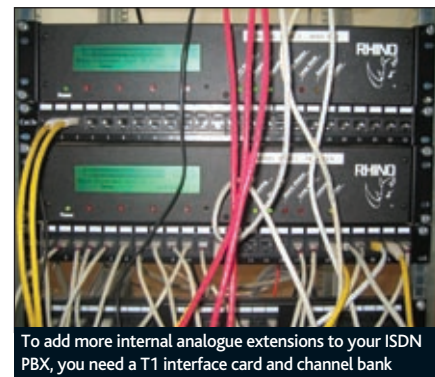
You may experience problems when using SIP with a Network Address Translation (NAT) firewall. If you intend to place your Asterisk PBX behind a NAT firewall, make sure that it is SIP aware, or that you can forward all the required ports. Linux's iptables packet filtering system can handle this. Ideally, you should avoid using NAT and assigning the Asterisk server with a public IP address. The ports that you will need to open and forward to your Asterisk server are SIP on UDP port 5060, IAX2 on UDP port 4569 and RTP on UDP ports 10,000–20,000.

As you can see from the small additions we have made to the dialplan (extensions.conf), it is very easy to set up a more complex system that can route calls based on their destination. For example, by sending international calls through an SIP provider you can take advantage of better overseas rates, while routing local calls through your analogue phone line could be useful when using Friends and Family-style deals.

The dialplan in Asterisk is the main part of your configuration and so will become your biggest configuration file. There are many commands available that we have not covered here but these are listed in the documentation. There are plenty of examples online, too.

MOVING ON AND UP

If you want a few more external lines for your Asterisk PBX, you could upgrade to an ISDN service. BT offers a two-channel (ISDN2e) and a 30-channel (ISDN30e) service, although with the latter you can subscribe to just the channels you need from a



To add more internal analogue extensions to your ISDN PBX, you need a T1 interface card and channel bank

minimum of eight. With these services you can assign more Direct Inward Dialling (DDI/DID) numbers than you have lines. For example, you may have an office with 30 staff but find that only eight of them will be on the phone at any one time. In this case you can get an ISDN30e circuit with eight channels but 30 DDI numbers, which would enable each member of staff to have their own number.

In addition, you may wish to have a lot more analogue extensions on the inside of your phone system. The best way to do this is to get a T1 interface card from Digium. This is the US version of the ISDN30 but with only 24 channels. You'd then need to use a T1 channel bank. We've had some experience of channel banks from Rhino Corp (see www.channelbanks.com). They are self-configuring and so no in-depth knowledge of telephony equipment is needed. With this setup, you simply have a single cable coming from your phone server to the channel bank, and from there you have 24 analogue lines available for phones or fax machines.

You'll find a wealth of information about Asterisk on the internet, but the main resource is at www.voip-info.org. This is an interactive site that's maintained by users from around the world. It contains all the command and file definitions and manuals, as well as caveats and examples that everyday users have encountered and fixed, and workarounds they have found. If you would like to take your Asterisk knowledge further, this is definitely the place to go.

Mailing lists are also available, along with archives, at www.asterisk.org/support. You can chat with people using Internet Relay Chat (IRC) on irc.freenode.net. Just log on to the #asterisk channel. Digium also offers commercial support and installation services for Asterisk. If you're installing Asterisk in a commercial environment this might make you feel more at ease, and it can be a godsend if the system goes down while your administrator is on holiday.

Asterisk is a very powerful phone system that easily rivals offerings from the likes of Alcatel, Avaya and Nortel. Whether you run a small office or a multinational with multiple sites and thousands of extensions and call centres, Asterisk can be adapted to handle the challenge and will probably save your company a large amount of money. If you need a new phone system, Asterisk is for you. **CS**

NEXT MONTH

LINUX TIPS

We give you a wealth of useful tips and tricks to help you wring every last drop of usefulness out your Linux system